Making Hybrid Tsunami Simulators in a Parallel Software Framework

Xing Cai[1,2] and Hans Petter Langtangen[1,2]

[1] Simula Research Laboratory, P.O. Box 134, NO-1325 Lysaker, Norway
{xingca, hpl}@simula.no
[2] Department of Informatics, University of Oslo, P.O. Box 1080 Blindern, NO-0316 Oslo, Norway

Abstract. Extending the philosophy of additive Schwarz algorithms, we propose a hybrid framework that allows different subdomains to use different mathematical models, different spatial discretizations, different local mesh types, and even different serial codes. This hybrid software framework is implemented using object-oriented techniques, such that existing serial codes are easily reused after being equipped with the standard interface of a generic subdomain solver. The resulting hybrid parallel tsunami simulator thus has full flexibility and extensibility. The focus of this paper is on the software design of the framework, with an illustrating example of application.

1 Introduction and Motivation

Computing the propagation of waves in the open sea is a key issue in tsunami simulation. When an entire ocean is the solution domain, this computational task becomes extremely challenging, both due to the huge amount of computations needed and due to the fact that different physics are valid in different regions. For example, the effect of dispersion is important for modeling wave propagation over an vast region with large water depth (see e.g. [5,10]). Moreover, in regions where water depth rapidly changes or close to the coastlines, nonlinear effects become important. Both the above factors mean that using a simple wave propagation model (such as the shallow water formulation (3)-(4) to be given later) over an entire ocean domain may seriously affect the accuracy (see e.g. [8,6]). Therefore, the computations should preferably adopt a hybrid strategy, i.e., using advanced mathematical models in small local areas where needed, while applying simple models to the remaining large regions. This is for achieving an acceptable balance between computational efficiency and accuracy.

Although tsunami simulations often employ the non-dispersive standard shallow water equations, we have in this work applied a set of Boussinesq equations, which are capable of modeling weakly dispersive and nonlinear waves:

$$\frac{\partial \eta}{\partial t} + \nabla \cdot (H + \alpha\eta)\nabla\phi + \epsilon H \left(\frac{1}{6}\frac{\partial \eta}{\partial t} - \frac{1}{3}\nabla H \cdot \nabla\phi \right) \nabla H = 0, \tag{1}$$

$$\frac{\partial \phi}{\partial t} + \frac{\alpha}{2}\nabla\phi \cdot \nabla\phi + \eta - \frac{\epsilon}{2} H \nabla \cdot \left(H \nabla \frac{\partial \phi}{\partial t} \right) + \frac{\epsilon}{6} H^2 \nabla^2 \frac{\partial \phi}{\partial t} = 0, \tag{2}$$

B. Kågström et al. (Eds.): PARA 2006, LNCS 4699, pp. 686–693, 2007.

Equation (1) is called the continuity equation, and Equation (2) is a variant of the Bernoulli (momentum) equation. In the above equations, η and ϕ are the primary unknowns denoting, respectively, the water surface elevation and the velocity potential. The water depth H is assumed to be a function of the spatial coordinates x and y. In (1)-(2) the effect of dispersion and nonlinearity is controlled by the two dimensionless constants ϵ and α, respectively. For more mathematical and numerical details, we refer to [10,7,3]. Note that by choosing $\epsilon = \alpha = 0$, we recover the widely used linear shallow water equations:

$$\frac{\partial \eta}{\partial t} + \nabla \cdot (H \nabla \phi) = 0, \tag{3}$$

$$\frac{\partial \phi}{\partial t} + \eta = 0. \tag{4}$$

The Boussinesq equations (1)-(2) can be considered a compromise between the computationally too expensive Navier-Stokes equations and the simple shallow water model (3)-(4). The numerical algorithm for solving (1)-(2) typically consists of a time-stepping process that solves the following two semi-discretized equations per time level:

$$\frac{\eta^{\ell} - \eta^{\ell-1}}{\Delta t} + \nabla \cdot \left(\left(H + \alpha \frac{\eta^{\ell-1} + \eta^{\ell}}{2} \right) \nabla \phi^{\ell - \frac{1}{2}} + \epsilon H \left(\frac{1}{6} \frac{\eta^{\ell} - \eta^{\ell-1}}{\Delta t} - \frac{1}{3} \nabla H \cdot \nabla \phi^{\ell - \frac{1}{2}} \right) \nabla H \right) = 0, \tag{5}$$

$$\frac{\phi^{\ell+\frac{1}{2}} - \phi^{\ell-\frac{1}{2}}}{\Delta t} + \frac{\alpha}{2} \nabla \phi^{\ell-\frac{1}{2}} \cdot \nabla \phi^{\ell+\frac{1}{2}} - \frac{\epsilon H}{2} \nabla \cdot \left(H \frac{\nabla \phi^{\ell+\frac{1}{2}} - \nabla \phi^{\ell-\frac{1}{2}}}{\Delta t} \right) + \frac{\epsilon H^2}{6} \frac{\nabla^2 \phi^{\ell+\frac{1}{2}} - \nabla^2 \phi^{\ell-\frac{1}{2}}}{\Delta t} = -\eta^{\ell}. \tag{6}$$

The above numerical scheme has adopted centered differences in the temporal direction and an associated staggered temporal grid [9]. The superscript ℓ in (5)-(6) denotes the time level. For the spatial discretization, both finite elements and finite differences can be used, depending on whether or not unstructured (and adaptively refined) meshes are needed to resolve the details of the water depth and/or the shape of coastlines. We mention that the shallow water model (3)-(4), being a special case of the Boussinesq equations (1)-(2), can be discretized in the temporal direction in the same fashion as in (5)-(6), likewise for the subsequent spatial discretization. The difference is that the resulting numerical strategy for solving (3)-(4) is often of an explicit nature (no need to solve linear systems), giving rise to an extremely fast algorithm. In contrast, the numerical strategy for (1)-(2) is of an implicit nature, meaning that linear systems must be solved for both (5) and (6) at every discrete time level. Moreover, unstructured finite element meshes will incur more computation time, in comparison with solving linear systems related to (5)-(6) on uniform finite difference spatial meshes. This is due to the complex data structure and indirect memory access that are used by the finite element codes. Therefore, in respect of software, the coding complexity

and computational cost also suggest that advanced mathematical models and unstructured computational meshes should only be applied to local small areas where necessary.

2 Parallelization by a Subdomain-Based Approach

Parallel computing is essential for simulating wave propagation over an entire ocean, because a huge number of degrees of freedom are often needed. As we have discussed above, different physics are valid in different regions, calling for a computationally resource-aware parallelization. More specifically, in regions where nonlinear and/or dispersive effects are important, existing serial software for Boussinesq equations (1)-(2) should be applied. Likewise can existing serial software for linear shallow water equations (3)-(4) be used in the remaining regions.

Such a parallelization strategy is most easily realized by using subdomains, such that the entire spatial domain Ω is decomposed into a set of overlapping subdomains $\{\Omega_s\}_{s=1}^P$. Mathematically, this idea of parallelization was first conceived in the additive Schwarz algorithms, see [11]. In a generic setting, where a partial differential equation (PDE) is expressed as

$$\mathcal{L}_\Omega(u) = f_\Omega,$$

the Schwarz algorithm consists of an iterative process generating $u^0, u^1, \ldots, u^k$ as a series of approximate solutions. During Schwarz iteration k, each subdomain first *independently* updates its local solution through

$$\mathcal{L}_{\Omega_s}(u_s^k) = f_{\Omega_s}^{k-1}. \tag{7}$$

Note that notation $f_{\Omega_s}^{k-1}$ means a right-hand side due to restricting f_Ω within Ω_s while making use of the latest global approximation u^{k-1} on the internal boundaries of Ω_s. When all the subdomains have finished solving (7), the new global solution u^k is composed by "sewing together" the subdomain local solutions $u_1^k, u_2^k, \ldots, u_P^k$.

Equation (7) thus opens for the possibility of using different local solvers in different subdomains. Taking the idea of additive Schwarz one step further, we can also apply different mathematical models in different subdomains. Therefore, different serial codes may be deployed regionwise. In the context of solving (5)-(6), these two equation can each use a series of the above Schwarz iterations at every discrete time level. The same set of subdomains should be used for both (5) and (6), to avoid unnecessary cross-subdomain data shuffle.

3 An Object-Oriented Implementation

To implement a hybrid parallel tsunami simulator as argued above, we resort to object-oriented programming techniques. For simplicity, the implementation can

be extended from a generic library of Schwarz algorithms for solving PDEs, such as that described in [1]. Our objective is a flexible design, such that existing serial wave propagation codes can be easily integrated into a hybrid parallel simulator of wave propagation. We will use C++ syntax in the following text, but the object-oriented strategy is equally implementable using another language such as Python, see e.g. [2].

3.1 A Generic Schwarz Framework

Before explaining the overall design of an object-oriented hybrid parallel tsunami simulator, it is necessary to briefly repeat the generic library of Schwarz algorithms, as described in [1]. Let us assume that the generic Schwarz library consists of two generic components, say, `class SubdomainSolver` and `class Administrator`. The purpose of the generic base class `SubdomainSolver` is to declare on beforehand a generic interface of all concrete subdomain solvers, which can later be inserted into the generic Schwarz framework. The generic interface is namely a set of virtual member functions without concrete implementation. For example, `createLocalMatrix` is a virtual function meant for setting up the subdomain matrix associated with discretizing (7), and function `solveLocal` is meant for solving the discretized form of (7) during each Schwarz iteration. The actual computational work is of course realized inside a concrete subclass of `SubdomainSolver`, which implements the virtual member functions such as `createLocalMatrix` and `solveLocal`, either by a cut-and-paste of old serial codes or more elegantly as a wrapper of an existing solver class.

Regarding the generic base class `Administrator`, the purpose is also to implement on beforehand a common set of functions, some of them as virtual member functions, useful later in a concrete parallel PDE solver based on additive Schwarz iterations. The typical functions of `Administrator` deal with, e.g., checking the global convergence among subdomains and invoking required inter-subdomain communication, all of which are independent of specific PDEs.

3.2 Designing a Parallel Tsunami Simulator

Now it is time for us to present the design of a hybrid parallel tsunami simulator. To maintain flexibility, while considering the special features with solving the Boussinesq water wave equations (1)-(2) (recall that (3)-(4) is a special case), we introduce a new generic class `SubdomainBQSolver`. The class is derived as a subclass of `SubdomainSolver` to implement all the virtual member functions of `SubdomainSolver`, while introducing a small set of new virtual functions. This is because, e.g., the two semi-discretized equations (5)-(6) both need to be solved using additive Schwarz iterations, thus requiring the `solveLocal` function to contain two versions, one for (5) and the other for (6). The structure of function `SubdomainBQSolver::solveLocal` may therefore be as follows:

```
if (solve_4_continuity)
  return solveContinuityEq ();
else
  return solveBernoulliEq ();
```

Here, `solve_4_continuity` is a flag indicating which equation, (5) or (6), is the current solution target. We note that the two new virtual member functions `solveContinuityEq` and `solveBernoulliEq` are left open for concrete subclasses to later insert their actual code of computation.

Once the new generic class `SubdomainBQSolver` is ready, a new class with name `HybridBQSolver` is derived as a subclass from base class `Administrator`. All the virtual functions of `Administrator` are implemented in `HybridBQSolver`, which also implements the time-stepping process that solves (5)-(6) at each time level. This in turn relies on an object of `SubdomainBQSolver` to perform the actual subdomain work for solving (7) during each Schwarz iteration. We remark that an object of `HybridBQSolver` and an object of a subclass of `SubdomainBQSolver` will be deployed on each processor during a parallel simulation, and inter-subdomain communication is handled between the objects of `HybridBQSolver`. The remaining programming work needed to implement an actual parallel tsunami simulator is mainly in form of deriving concrete subclass(es) of `SubdomainBQSolver`, best illustrated by the following case study. A hybrid parallel simulator arises when objects of different subclasses of `SubdomainBQSolver` are deployed on different subdomains.

4 Case Study

We have two existing serial software codes: (1) an advanced C++ finite element solver named `class Boussinesq` applicable for unstructured meshes, and (2) a legacy F77 finite difference code applicable for uniform meshes. Both codes are hard to parallelize following the standard approach of inserting MPI commands directly into linear algebra operations. This is especially true for the legacy F77 code, which has a tangled internal data structure. Our objective is to build a hybrid parallel tsunami simulator based on these two codes, in a straightforward and effective way. To this end two light-weight new classes are programmed:

`class SubdomainBQFEMSolver` and `class SubdomainBQFDMSolver`

Here, class `SubdomainBQFEMSolver` uses double inheritance, as subclass of both `SubdomainBQSolver` and `Boussinesq`, so that it inherits the computational functionality from `Boussinesq` and at the same time is accepted by `HybridBQSolver` as a subdomain solver. Similarly, class `SubdomainBQFDMSolver` is derived from `SubdomainBQSolver` and at the same time "wraps up" the F77 subroutines of the legacy code inside its `solveLocal` function.

Using such a hybrid software framework, a parallel tsunami simulator has been built for the 2004 Indian Ocean tsunami. The entire spatial domain is depicted in Fig. 1, where the epicenter is located at position $(1, 1)$. Moreover,

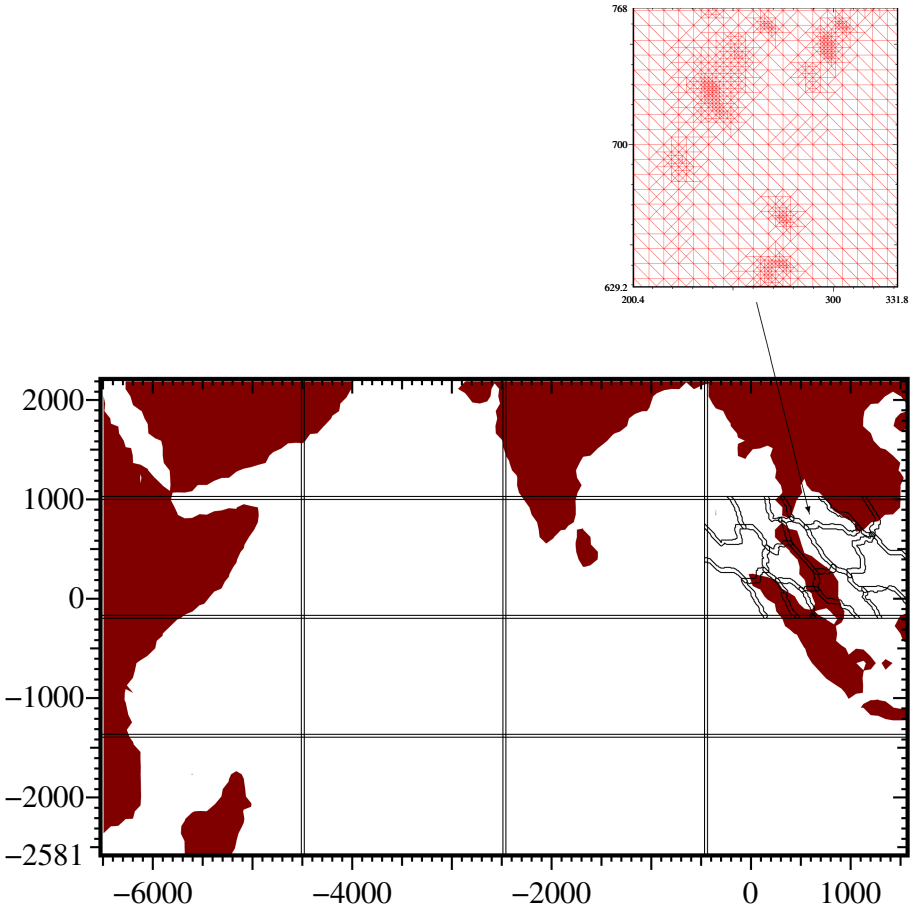

Fig. 1. An example of partitioning the Indian Ocean domain into a mixture of rectangular and complex-shaped subdomains. Finite differences are used by the rectangular subdomains to carry out the spatial discretization, whereas the complex-shaped subdomains use finite elements and adaptively refined local meshes.

the figure also shows different types of local meshes, i.e., uniform local meshes in the rectangular subdomains and adaptively refined local meshes in the complex-shaped subdomains. Different spatial discretizations (finite differences and finite elements) can thus be deployed in different regions. The simulation results have been reported in [4].

5 Concluding Remarks

We have explained a hybrid software framework for parallelizing and, at the same time, combining different existing serial codes. Such a parallelization strat-

egy is numerically inspired by the additive Schwarz algorithms, while implementationally enabled by object-oriented programming techniques. The approach is particularly attractive for creating parallel simulators of wave propagation, as many old serial wave codes exist but are otherwise difficult to be parallelized. For ocean-scale simulations, the advantage of such a hybrid parallel simulator is that small areas of difficulty can be handled by subdomains that are equipped with an advanced mathematical model and a sophisticated numerical solver, whereas the remaining vast regions are handled by a simple mathematical model and fast code.

Future work will apply the software approach from this paper to other aspects of tsunami simulation, for instance, run-up of waves on beaches. Quite some sophisticated serial codes have been developed for the run-up problem, and these are hard to parallelize well. Our suggested approach makes parallelization feasible with little work. Of even more importance is the fact that reuse of very well-tested codes contributes to high reliability in a new hybrid, parallel simulator. For each new problem such as wave run-up it always remains, however, to investigate whether the additive Schwarz algorithm is capable of delivering satisfactory parallel efficiency.

References

1. Cai, X.: Overlapping domain decomposition methods. In: Langtangen, H.P., Tveito, A. (eds.) Advanced Topics in Computational Partial Differential Equations – Numerical Methods and Diffpack Programming, pp. 57–95. Springer, Heidelberg (2003)
2. Cai, X., Langtangen, H.P.: Parallelizing PDE solvers using the Python programming language. In: Bruaset, A.M., Tveito, A. (eds.) Numerical Solution of Partial Differential Equations on Parallel Computers, Lecture Notes in Computational Science and Engineering, vol. 51, pp. 295–325. Springer, Heidelberg (2006)
3. Cai, X., Pedersen, G.K., Langtangen, H.P.: A parallel multi-subdomain strategy for solving Boussinesq water wave equations. Advances in Water Resources 28, 215–233 (2005)
4. Cai, X., Pedersen, G.K., Langtangen, H.P., Glimsdal, S.: Parallel simulation of tsunamis using a hybrid software approach. In: Joubert, G.R., et al. (eds.) Parallel Computing: Current & Future Issues of High-End Computing; Proceedings of the International Conference ParCo 2005, pp. 383–390. John von Neumann Institute for Computing (2006)
5. Gjevik, B., Pedersen, G., Dybesland, E., Harbitz, C.B., Miranda, P.M.A., Baptista, M.A., Mendes-Victor, L., Heinrich, P., Roche, R., Guesmia, M.: Modeling tsunamis from earthquake sources near Gorringe Bank southwest of Portugal. J. Geophysical Research 102, 931–949 (1997)
6. Glimsdal, S., Pedersen, G., Atakan, K., Harbitz, C.B., Langtangen, H.P., Løvholt, F.: Propagation of the Dec. 26, 2004, Indian Ocean Tsunami: Effects of dispersion and source characteristics. Int. J. Fluid Mech. Research 33, 15–43 (2006)
7. Glimsdal, S., Pedersen, G.K., Langtangen, H.P.: An investigation of overlapping domain decomposition methods for one-dimensional dispersive long wave equations. Advances in Water Resources 27, 1111–1133 (2004)

8. Okal, E.A., Synolakis, C.E.: Source discriminants for near-field tsunamis. Geophysical Journal International 158(3), 899–912 (2004)
9. Pedersen, G.: Three-dimensional wave patterns generated by moving disturbances at transcritical speeds. J. Fluid Mech. 196, 39–63 (1988)
10. Pedersen, G., Langtangen, H.P.: Dispersive effects on tsunamis. In: Proceedings of the International Conferance on Tsunamis, Paris, France, pp. 325–340 (1999)
11. Smith, B.F., Bjørstad, P.E., Gropp, W.: Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations. Cambridge University Press, Cambridge (1996)